ELSEVIER

SciVerse ScienceDirect
Disponible en www.sciencedirect.com

Revista Iberoamericana de Automática e Informática industrial 9 (2012) 347–358

RIAI
www.elsevier.es/RIAI

Algoritmos *Wavenet* con Aplicaciones en la Aproximación de Señales: un Estudio Comparativo

C. R. Domínguez Mayorga[a], M. A. Espejel Rivera[b], L. E. Ramos Velasco[a,c,*], J. C. Ramos Fernández[c], E. Escamilla Hernández[d]

[a] *Universidad Autónoma del Estado de Hidalgo, CIMA-ICBI, CITIS-ICBI, Carretera Pachuca-Tulancingo, Km. 4.5, Mineral de la Reforma, Hidalgo, México.*
[b] *Universidad la Salle Pachuca, Campus La Concepción, Av. San Juan Bautista de La Salle No. 1. San Juan Tilcuautla, San Agustín Tlaxiaca, Hidalgo, México.*
[c] *Universidad Politécnica de Pachuca, Carretera Pachuca-Cd. Sahagún, Km. 20, Rancho Luna, Ex-Hacienda de Sta. Bárbara, Zempoala, Hidalgo, México.*
[d] *SEPI-ESIME Cul. IPN, Av. Santa Ana 1000, Col. San Francisco Culhuacan, Del. Coyoacan, México.*

Resumen

En este trabajo de investigación se aplican métodos adaptables en el diseño de algoritmos computacionales, dichos algoritmos emplean redes neuronales y series de *wavelets* para construir "neuroaproximadores" *wavenets*. Se muestra cómo las *wavenets* pueden combinarse con los métodos autosintonizables para obtener el seguimiento de señles complejas que están en función del tiempo. Los algoritmos obtenidos se aplican en la aproximación de señales que representan funciones algebraicas y funciones aleatorias, así como en una señal médica deun ECG. Se muestran los resultados en simulación numérica de dos arquitecturas de neuroaproximadores *wavenets*: el primero está basado en una *wavenet*, con el cual se aproximan las señales bajo estudio donde los parámetros de la red neuronal son ajustados en línea; el otro esquema emplea un filtro IIR a la salida de la red *wavenet* para discriminar las contribuciones de aquellas neuronas que tienen menos peso en la aproximación de la señal, lo que ayuda a reducir el tiempo de convergencia a un error mínimo deseado.

Palabras Clave:
Procesamiento de señales, Algoritmos auto-ajustables, Redes neuronales, Algoritmos de aproximación, Método del gradiente.

1. Introducción

La contribución de las ciencias computacionales en el desarrollo de nuevos y mejores algoritmos para la aproximación de señales en diferentes áreas de la ingeniería, hoy en día es una realidad (Widrow and Stearns, 1985; Haykin, 1994, 2001; Gupta et al., 2003). Este avance permite la implantación de algoritmos computacionales cada vez más complejos, tal es el caso de algoritmos que hacen uso de redes neuronales, lógica difusa como se muestra en (Park and Sandberg, 1991; Jun and Hong, 2005; Ting and Sugai, 1999; Li and Chen, 2002b; Yu and Li, 2003; W. Wang and Wang, 1997). En (Kobayashi and Torioka, 1994) se hace uso de redes neuronales *wavelets* para realizar aproximación de funciones, dividiendo el trabajo en dos partes: en la primera parte se propone una red de tres capas (1,*N*,1), donde las neuronas de la entrada y salida son elementos lineales y la función de activación de la capa oculta es una *wavelet* madre, obteniendo gradualmente el número de neuronas (*N*) requeridas para cubrir la región de tiempo y frecuencia de una función objetivo, mientras se van calculando los parámetros de traslación y escalamiento de la *wavelet*, así como los pesos de cada neurona usando la regla de Kohonen, en la segunda parte se realiza la minimización de los errores usando el algoritmo de retropropagación localizado. Además, se hace una comparación de los resultados obtenidos contra los producidos por una red de retropropagación tradicional, presentando una convergencia más rápida el algoritmo propuesto y mejores resultados tanto para datos de entrenamiento como para datos de prueba.

Una diferencia notable entre el algoritmo propuesto en (Kobayashi and Torioka, 1994) y el que se propone en este artículo es que en (Kobayashi and Torioka, 1994) se divide el algoritmo en dos partes lo cual retarda la convergencia con respecto a la del algoritmo propuesto en este artículo, ésto debido a que primero debe de obtenerse el número suficiente de neuronas para cubrir la región de la función objetivo y posteriormente se comienza con la minimización del error, mientras que el algoritmo aquí propuesto presenta un filtro IIR a la salida de la red *wavelet*

* Autor en correspondencia.
Correos electrónicos: carlosrdm@hotmail.com (C. R. Domínguez Mayorga), aespejel@lasallep.edu.mx (M. A. Espejel Rivera), lramos@uaeh.edu.mx (L. E. Ramos Velasco), jramos@upp.edu.mx (J. C. Ramos Fernández)
URL: www.lasallep.edu.mx/ (M. A. Espejel Rivera), http://www.upp.edu.mx/posgrado/mecatronica/ (J. C. Ramos Fernández)

http://dx.doi.org/10.1016/j.riai.2012.09.001

el cual hace las veces de la optimización de la red (que en este caso discrimina neuronas en lugar de agregarlas) durante el proceso de minimización. Una de las aplicaciones de los resultados presentados en el artículo se dan en la identificación y control de plantas estables como se muestran en (Cruz-Tolentino et al., 2010; Sedighizadeh and Rezazadeh, 2008; Islas-Gómez et al., 2010, 2012).

En (Li and Chen, 2002a) se hace uso de una red neuronal *wavelet* robusta basada en la teoría de la regresión robusta, aplicada en el marco de la aproximación de funciones, ajustando adaptativamente el número de datos de entrenamiento involucrados durante el entrenamiento. Para mejorar la robustez de las redes *wavenets*, el procedimiento de entrenamiento para los parámetros iniciales es llevado a cabo por el algoritmo LTS (*least trimmer square*) descrito en (Rousseeuw and Leroy, 1987). En este caso se muestran dos ejemplos de aproximación, uno en funciones de una dimensión y el otro para una función de dos dimensiones.

A diferencia del algoritmo LTS propuesto en (Li and Chen, 2002a) para el entrenamiento de la red, el que se implementa en este artículo (el LMS) no requiere de una manipulación previa de los errores (el algoritmo LTS realiza un ordenamiento de los errores antes de comenzar), lo cual podría retardar el funcionamiento del algoritmo.

En (Chen and Hui-Qiang, 2007) se propone un método para implementar el convertidor analógico digital (ADC) de alta precisión usando una red neuronal *wavelet* para aproximar y compensar la no linealidad del ADC. La red propuesta es un red de 3 capas, en donde la capa de salida implementa la función sigmoide, mientras que la *wavelet* madre Morlet es implementada en el resto de la red. Este tipo de red requiere de un número pequeño de iteraciones y parámetros en comparación con los perceptrones multicapa. El algoritmo propuesto en (Chen and Hui-Qiang, 2007) es muy similar al de este artículo, solo que la salida en este caso es la función identidad y en el presentado en (Chen and Hui-Qiang, 2007) es la función sigmoide, la cual acota la salida en el intervalo (01), que muestra ser útil para el caso que se estudia. Otra diferencia con el algoritmo aquí propuesto es que no se implementa el filtro IIR a la salida de la red.

En (S. Gopinath and Bhatt, 2004) se propone una red neuronal *wavelet* para el problema de identificación en línea, proponiendo un esquema de identificación basado en redes neuronales wavelets, así como un algoritmo de aprendizaje para la identificación en línea de sistemas no lineales. Ésto presenta algunas técnicas que se podrían implementar en trabajos futuros al extender el algoritmo propuesto en este artículo a problemas en tiempo real.

En (Sitharama et al., 2002) su presentan los fundamentos de las redes *wavenets*, así como algunas de sus aplicaciones. Se presenta un esquema de aprendizaje recurrente, en redes *wavelets* dinámicas (el cual es muy parecido al propuesto en este artículo sin el IIR en la salida de la red), este tipo de aprendizaje presenta buenos resultados para simulación numérica en la aproximación de funciones. Se hace también una comparación con respecto a las redes neuronales de base radial (las cuales son ya buenos aproximadores de funciones) presentando varias ventajas sobre éstas.

Una de las aplicaciones importantes que se describen en (Sitharama et al., 2002) es la predicción de series de tiempo caóticas como, por ejemplo, el atractor de Ikeda, el atractor de Lorenz. También se presentan diferentes variantes de las técnicas de aprendizaje empleada. En (Sitharama et al., 2002) se presentan la teoría básica sobre las *wavenets* que se implementan en este artículo, así como algunas variantes del algoritmo de entrenamiento aquí implantado, lo que fue de gran utilidad en el desarrollo de éste.

El artículo está organizado de la siguiente manera: una descripción general de redes neuronales *wavelets* es dada en la Sección 2, mostrando la primera arquitectura *wavenet* utilizada en este artículo. En la Sección 3 se presenta la aproximación de señales mediante una *wavenet*. En la Sección 4 se presenta la segunda arquitectura de la red *wavenet* con filtro IIR, mostrando en la Sección 5 los resultados de las simulaciones numéricas. La comparación entre las dos arquitecturas empleadas se muestran en la Sección 6. En la Sección 7 se dan los comentarios sobre los resultados de las dos arquitecturas empleadas. Finalmente, las conclusiones sobre los resultados obtenidos se presentan en la Sección 8.

2. Redes Neuronales Wavelets Adaptables (*Wavenets*)

Combinando la teoría de la transformada *wavelet* con el concepto básico de redes neuronales, se propone un nuevo mapeo de red llamado red neuronal *wavelets* adaptable (WN) o *wavenets* como una alternativa a las redes neuronales realimentadas para aproximar arbitrariamente funciones no lineales (Zhang and Beneveniste, 1992). Los algoritmos *wavenet* consisten básicamente de dos procesos: la auto-construcción de las redes y la minimización del error de aproximación. En el primer proceso, las estructuras de las redes aplicadas para la representación son determinadas usando análisis *wavenet*. La red gradualmente combina unidades ocultas para cubrir eficiente y suficientemente la región tiempo-frecuencia ocupada por una meta dada. Simultáneamente, los parámetros de la red son actualizados para conservar la topología de la red y aprovechar el proceso posterior. En el segundo proceso, las aproximaciones de los errores instantáneos son minimizadas usando una técnica de adaptación basada en el algoritmo del gradiente descendente, es decir los parámetros de la red inicializada son actualizados usando este método. Cada unidad oculta tiene una ventana cuadrada en el plano tiempo-frecuencia. La regla de optimización solamente se aplica a las unidades ocultas donde el punto seleccionado cae en sus ventanas. Por lo tanto, el costo del aprendizaje puede ser reducido. Todas estas ventajas que se tienen de las redes *wavenets* son aprovechadas en el aproximadores *wavenet* que se emplea en este trabajo de investigación.

La arquitectura de las *wavenets* mostrada en la Figura 1 aproxima la señal deseada $u(t)$ mediante la generalización de una combinación lineal de un conjunto de *wavelets* hijas $h_{a,b}(t)$, donde éstas son generadas por una dilatación a y una traslación b de la *wavelet* madre $h(t)$ (Chui, 1992; Daubechies, 1992;

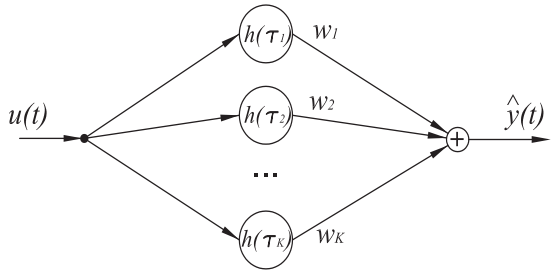


Figura 1: Arquitectura de la red *wavenet* empleada, en la cual $\tau_k = \frac{t-b_k}{a_k}$, es el argumento de la k-ésima *wavelet* hija, donde $k = 1, 2, ..., K$.

Mallat, 1995; Teolis, 1998; Vetterli and Kovačević, 1995):

$$h_{a,b}(t) = h\left(\tfrac{t-b}{a}\right) \tag{1}$$

con el factor de dilatación $a \neq 0, b \in \mathbb{R}$.

La señal aproximada de la red $\hat{y}(t)$ puede ser representada por:

$$\hat{y}(t) = u(t) \sum_{k=1}^{K} w_k h_{a_k,b_k}(t) \tag{2}$$

donde K es el número de ventanas *wavelets*, w_k son los pesos y $h(t)$ es una *wavelet* madre, a_k y b_k son el escalamiento y la traslación respectivamente de la k-ésima neurona. En la Tabla 1 se dan las *wavelets* seleccionadas para este artículo empleadas para la aproximación de las señales.

Tabla 1: *Wavelets* madres implementadas en las *Wavenets*, donde $\tau = \frac{t-b}{a}$.

Wavelet Madre	$h(\tau)$
Morlet	$cos(\omega_0\tau)e^{-0.5\tau^2}$
RASP1	$\frac{\tau}{(\tau^2+1)^2}$
RASP2	$\frac{\tau cos(\tau)}{\tau^2+1}$
RASP3	$\frac{sen(\pi\tau)}{\tau^2-1}$
SLOG1	$\frac{1}{1+e^{-\tau+1}} - \frac{1}{1+e^{-\tau+3}} - \frac{1}{1+e^{-\tau-3}} + \frac{1}{1+e^{-\tau-1}}$
SLOG2	$\frac{3}{1+e^{-\tau-1}} - \frac{3}{1+e^{-\tau+1}} - \frac{1}{1+e^{-\tau-3}} + \frac{1}{1+e^{-\tau+3}}$
POLYWOG1	$\tau e^{-\frac{\tau^2}{2}}$
POLYWOG2	$(\tau^3 - 3\tau)e^{-\frac{\tau^2}{2}}$
POLYWOG3	$(\tau^4 - 6\tau^2 + 3)e^{-\frac{\tau^2}{2}}$
POLYWOG4	$(1 - \tau^2)e^{-\frac{\tau^2}{2}}$
POLYWOG5	$(3\tau^2 - \tau^4)e^{-\frac{\tau^2}{2}}$
Shannon	$\frac{sen(2\pi\tau)-sen(\pi\tau)}{\pi\tau}$

Para calcular los gradientes empleados en reglas de actualización de los parámetros, se define la función de energía como:

$$E = \frac{1}{2} \sum_{t=1}^{T} e^2(t) \tag{3}$$

donde $e(t)$ representa el error de aproximación con respecto a una función objetivo $u(t)$ y la salida de la red $\hat{y}(t)$, definido como:

$$e(t) = u(t) - \hat{y}(t) \tag{4}$$

El objetivo es minimizar $E(w_k, a_k, b_k)$, variando los parámetros w_k, a_k y b_k, donde $k = 1, 2, \ldots, K$. Para ello se calculan los gradientes:

$$\frac{\partial E}{\partial w_k} = -\sum_{t=1}^{T} e(t)u(t)h(\tau_k) \tag{5}$$

$$\frac{\partial E}{\partial b_k} = -\sum_{t=1}^{T} e(t)u(t)w_k \frac{\partial h(\tau_k)}{\partial b_k} \tag{6}$$

$$\frac{\partial E}{\partial a_k} = -\sum_{t=1}^{T} e(t)u(t)w_k\tau_k \frac{\partial h(\tau_k)}{\partial b_k} = \tau_k \frac{\partial E}{\partial b_k} \tag{7}$$

Los incrementos de cada coeficiente son los negativos de sus gradientes,

$$\Delta \boldsymbol{w} = -\frac{\partial E}{\partial \boldsymbol{w}}, \quad \Delta \boldsymbol{a} = -\frac{\partial E}{\partial \boldsymbol{a}}, \quad \Delta \boldsymbol{b} = -\frac{\partial E}{\partial \boldsymbol{b}} \tag{8}$$

Así los coeficientes $\boldsymbol{w}$, $\boldsymbol{a}$ y $\boldsymbol{b}$ de la red *wavenet* son actualizados de acuerdo a las reglas

$$\boldsymbol{w}(t+1) = \boldsymbol{w}(t) + \mu_{\boldsymbol{w}} \Delta \boldsymbol{w} \tag{9}$$

$$\boldsymbol{a}(t+1) = \boldsymbol{a}(t) + \mu_{\boldsymbol{a}} \Delta \boldsymbol{a} \tag{10}$$

$$\boldsymbol{b}(t+1) = \boldsymbol{b}(t) + \mu_{\boldsymbol{b}} \Delta \boldsymbol{b} \tag{11}$$

donde μ es un parámetro fijo que ayuda a mejorar la rapidez de aprendizaje de la red *wavenet*, que se determina a prueba y error.

Algoritmo 1. El algoritmo *wavenet* que resulta de la *wavenet* es:

1. Calcular las salidas de la *wavenet* $\hat{y}(t)$ como en (2) para $t = 1, 2, ..., T$, es decir, una iteración (época).
2. Para cada uno de los valores de t calcular el error $e(t)$ con respecto a la señal de entrada $u(t)$ definido en (4).
3. Obtener la función de energía del error E definida en (3) y calcular $\frac{\partial E}{\partial w_k}$, $\frac{\partial E}{\partial a_k}$ y $\frac{\partial E}{\partial b_k}$ dadas en (5), (6) y (7) respectivamente.
4. Definir los incrementos $\Delta \boldsymbol{w}$, $\Delta \boldsymbol{a}$ y $\Delta \boldsymbol{b}$ para los parámetros $\boldsymbol{w}$, $\boldsymbol{a}$ y $\boldsymbol{b}$ como en (8).
5. Se realizan las actualizaciones de los parámetros $\boldsymbol{w}$, $\boldsymbol{a}$ y $\boldsymbol{b}$ de acuerdo con (9), (10) y (11), respectivamente.
6. Repetir el procedimiento el número de iteraciones (épocas) necesarias para que el error sea mínimo o alcance algún umbral dado $\varepsilon > 0$.

3. Aproximación de Señales Mediante una *Wavenet*

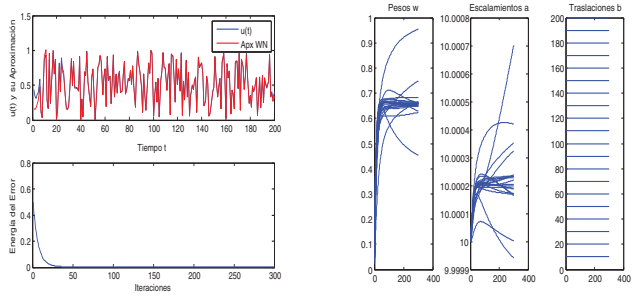
A continuación se hace un breve análisis de la aproximación de algunas funciones mediante las *wavenets*, haciendo comparaciones cuando se cambia el número de neuronas usadas y la *wavelet* implementada, manteniendo en todos los casos un umbral mínimo de error aceptado, $\varepsilon = 0.001$. Los valores iniciales de los parámetros empleados para el aprendizaje están dados en la Tabla 2 y son seleccionados a prueba y error.

Tabla 2: Valores iniciales propuestos de los parámetros de la *wavenet*.

μ_w	μ_a	μ_b	w_k	a_k	b_k	ε
0.01	0.001	0.0001	0	10	$\frac{T}{K}k$	0.001

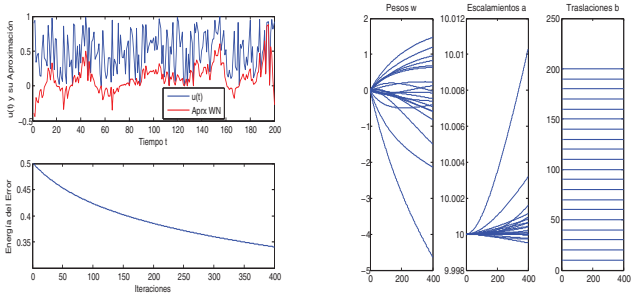
3.1. *Comparación Entre la Implementación de Diferentes Wavelets en la Aproximación de Señales*

En seguida se compara el comportamiento de la aproximación *wavenet* con una señal aleatoria y acotada, generada por la función *random* de MATLAB. Para esto se implementan en la red las *wavelets* dadas en la Tabla 1 con $K = 20$, aplicando el algoritmo de minimización de la función de energía del error hasta un máximo de 400 iteraciones o un umbral de error $\varepsilon = 0.001$ en cada uno de los casos.



(a) (b)

Figura 2: Aproximación *Wavenet* con 20 Neuronas y *Wavelet* Morlet, y su Función de Energía (a). La Actualización de los Parámetros Durante el Proceso de Aprendizaje (b).

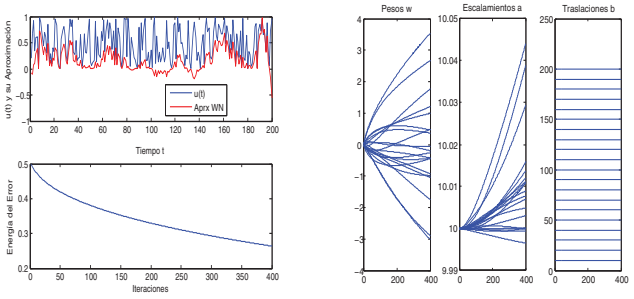


(a) (b)

Figura 3: Aproximación *Wavenet* con 20 Neuronas y *Wavelet* RASP1, y su Función de Energía (a). La Actualización de los Parámetros Durante el Proceso de Aprendizaje (b).

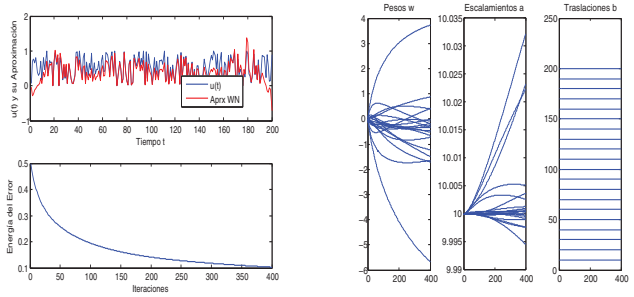
En la Figura 2 se muestra el comportamiento de la red *wavenet* implementando la *wavelet* Morlet, mostrando un desempeño aceptable al alcanzar el umbral de error alrededor de las 300 iteraciones y mostrando una buena convergencia desde antes de las 50 iteraciones.

Las Figuras 3 y 4, corresponden a la implementación de las *wavelets* RASP1 y RASP2 respectivamente, las cuales permiten



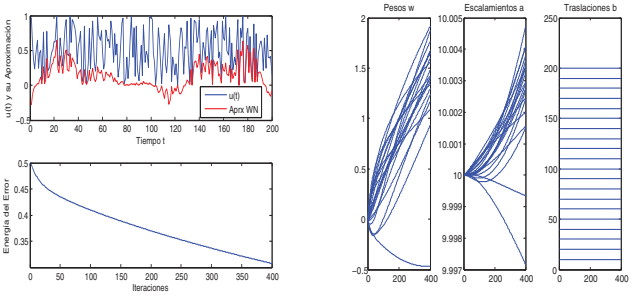
(a) (b)

Figura 4: Aproximación *Wavenet* con 20 Neuronas y *Wavelet* RASP2, y su Función de Energía (a). La Actualización de los Parámetros Durante el Proceso de Aprendizaje (b).



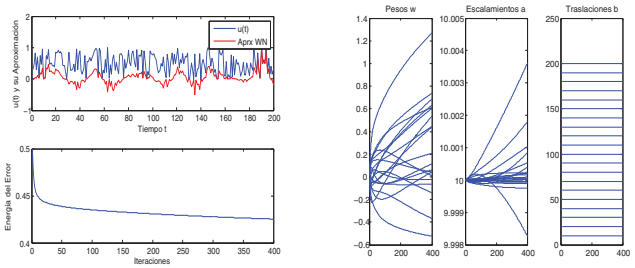
(a) (b)

Figura 5: Aproximación *Wavenet* con 20 Neuronas y *Wavelet* SLOG1, y su Función de Energía (a). La Actualización de los Parámetros Durante el Proceso de Aprendizaje (b).



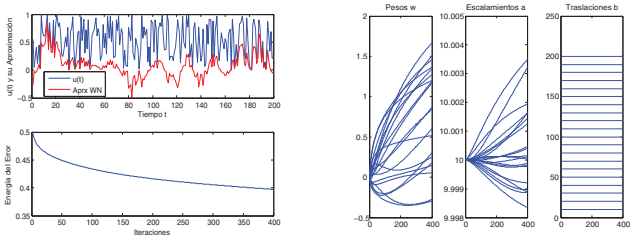
(a) (b)

Figura 6: Aproximación *Wavenet* con 20 Neuronas y *Wavelet* SLOG2, y su Función de Energía (a). La Actualización de los Parámetros Durante el Proceso de Aprendizaje (b).



(a) (b)

Figura 7: Aproximación *Wavenet* con 20 Neuronas y *Wavelet* POLYWOG1, y su Función de Energía (a). La Actualización de los Parámetros Durante el Proceso de Aprendizaje (b).



(a) (b)

Figura 8: Aproximación *Wavenet* con 20 Neuronas y *Wavelet* POLYWOG5, y su Función de Energía (a). La Actualización de los Parámetros Durante el Proceso de Aprendizaje (b).

observar que el desempeño de éstas no es tan bueno como el caso de la *wavelet* Morlet, observando que en las 400 iteraciones alcanzan valores del error < 0.350 y < 0.300 respectivamente, los cuales son altos en comparación con el umbral $\varepsilon = 0.001$.

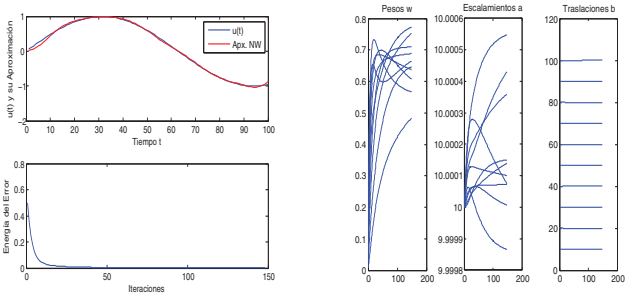
El desempeño de la red *wavenet* que se observa en las Figuras 5 y 6 corresponde a la implementación de las *wavelets* SLOG1 y SLOG2 respectivamente, mostrando un desempeño bajo en la aproximación, con un error alcanzado cercano al 0.100 y 0.300 respectivamente, esto al final de las 400 iteraciones.

Las Figuras 7 y 8 muestran el comportamiento de la aproximación *wavenet* al implementar las *wavelets* POLYWOG1 y POLYWOG5 respectivamente, mostrando un desempeño bajo alcanzando errores cercanos a 0.450 y 0.400 respectivamente al termino de las 400 iteraciones.

De lo anterior, se puede observar que para este tipo de funciones y esta configuración de la red, la *wavenet* que implementó a la *wavelet* madre Morlet es la que mejor se comportó en la aproximación, alcanzando valores de error bajos de manera muy rápida (menos de 50 iteraciones) y logrando el objetivo de alcanzar una aproximación con un umbral del error antes mencionado. Por lo tanto, en la siguiente subsección se emplea la *wavelet* Morlet para aproximar otro tipo de funciones cambiando el número de neuronas.

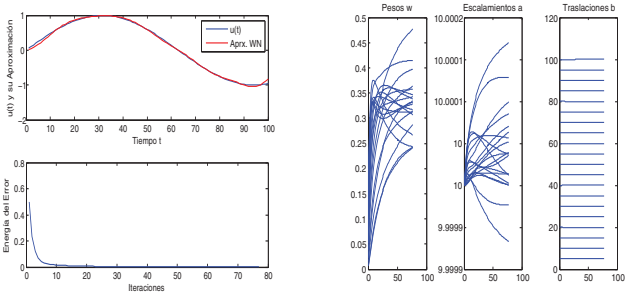
3.2. *Aproximación Wavenet con Wavelet Morlet y Diferentes Números de Neuronas*

La Figura 9 muestra los resultados de la aproximación de la función $u(t) = sen(t/20)$ en el intervalo $[1, 100]$ (con 100 muestras de $u(t)$) con una *wavenet* de $K = 10$ (neuronas) y la *wavelet* Morlet implementada, de la cual se puede observar en la Figura 9a que la función $u(t)$ es aproximada hasta alcanzar el umbral del error ε en poco menos de 150 iteraciones (épocas de entrenamiento), mientras que en la Figura 9b se observa el comportamiento de la actualización de los parámetros de la red (los pesos $\boldsymbol{w}$, los escalamientos $\boldsymbol{a}$ y las traslaciones $\boldsymbol{b}$) durante las iteraciones.



(a) (b)

Figura 9: Aproximación Wavenet con $K = 10$ y su Función de Energía (a). La Actualización de los Parámetros Durante el Proceso de Aprendizaje (b).



(a) (b)

Figura 10: Aproximación Wavenet con $K = 20$ y su Función de Energía (a). La Actualización de los Parámetros Durante el proceso de Aprendizaje (b).

La Figura 10 muestra los resultados de la aproximación de la misma función $u(t) = sen(t/20)$ en el intervalo $[1, 100]$ con una *wavenet* de $K = 20$ y la *wavelet* Morlet implementada, de la cual se puede observar en la Figura 10a que la función $u(t)$ es aproximada con un aceptable de error hasta alcanzar el umbral fijado de ε en menos de 80 iteraciones, mientras que en la Figura 10b se observa el comportamiento de la actualización de los parámetros de la red *wavenet*.

De estas dos Figuras 9 y 10 se observa que cuando se trabajó con una red con $K = 20$ existió una disminución de la función de energía más rápida (con respecto al número de iteraciones) que para el caso de la red con $K = 10$, a pesar de tener un número mayor de neuronas, la red con $K = 20$ requiere menor tiempo para alcanzar el umbral.

En la Tabla 3 se puede observar el comportamiento de las *wavenets* para diferentes valores de K, las columnas representan los números de cada columna con respecto al número de neuronas indicados por cada renglón, por ejemplo, el número de iteraciones = 9 que se encuentra en la intersección de la columna encabezada por 0.030 y el renglón encabezado por 15, corresponde al número de iteraciones requeridas para que una *wavenet* con $K = 15$ aproxime la función $u(t) = sen(t/20)$ hasta alcanzar un valor mínimo del error $e = 0.030$.

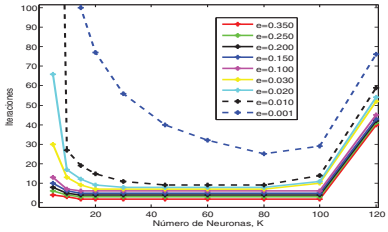


Figura 11: Iteraciones Requeridas por las *Wavenets* para Diferentes Valores de K con Respecto a Diferentes Umbrales de Error.

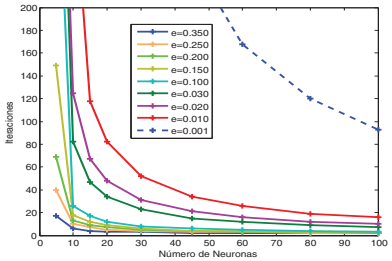
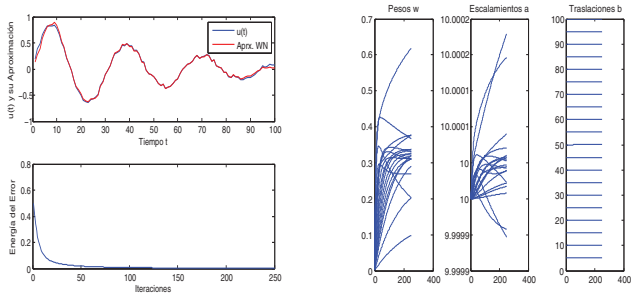


Figura 12: Iteraciones Requeridas por las *Wavenets* para Diferentes Valores de K con Respecto a Diferentes Umbrales de Error, para una señal de entrada $u(t) = exp(-t/50) * sin(t/5) + ruido$

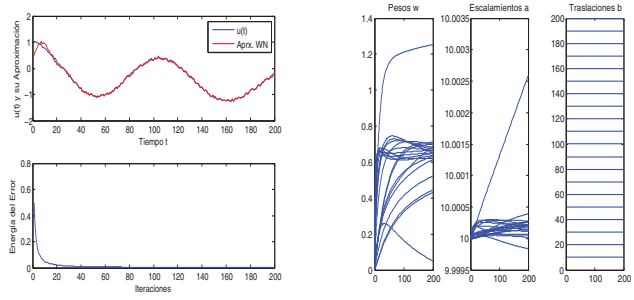
Los valores vistos en el Tabla 3 se muestran también en la Figura 11, por ejemplo, de ambas se observa que para un número de neuronas $K = 5$ en la *wavenet* se requieren 13 iteraciones para alcanzar el umbral del error $e = 0.100$, mientras que para ese mismo umbral una red con $K = 30$ requiere de 7 iteraciones y para una *wavenet* con $K = 100$ el número de iteraciones requerido es de 10. De ésto también se puede ver que la *wavenet* que realiza la aproximación de la señal $u(t) = sen(t/20)$ hasta tener un error $e = 0.001$ con un menor número de iteraciones posee un número de neuronas entre $K = 60$ y $K = 100$, de la misma manera se observa que para un número menor de neuronas (alrededor de $K = 5$) requiere un mayor número de iteraciones para alcanzar aproximaciones con valores de error mínimos, como para el caso de una cantidad considerable de neuronas (alrededor de $K = 100$), aunque para errores en el rango $0.1 \leq e \leq 0.5$ se requiere un número aproximado de iteraciones para las *wavenet* con los diferentes números de neuronas; por lo que para fines prácticos se compararon los comportamientos de diferentes *wavenets* con $K = 20$, solo cambiando la *wavelet* implementada para la aproximación de señales. Similarmente, se presenta en la Figura 12 las simulaciones numéricas para la señal $u(t) = exp(-t/50) * sin(t/5) + ruido$.

De las dos subsecciones anteriores se puede concluir que la *wavenet* que presentó un mejor desempeño es la que tiene las *wavelet* Morlet como funciones de activación con 20 neuronas, ya que el número de iteraciones que se requirieron para aproximar la señales bajo estudio fueron menores, así mismo el tiempo de procesamiento es menor. Por lo tanto a continuacion se empleará esta red *wavenet* Morlet para la aproximación de señales con ruido generado por la función *random* disponible en MATLAB.



(a) (b)

Figura 13: Aproximación *Wavenet* de la Función $u(t) = e^{-t/50} sin(\frac{t}{5}) + ruido$ con 20 Neuronas y *Wavelet* Morlet (a). La Actualización de los Parámetros Durante el proceso de Aprendizaje (b).



(a) (b)

Figura 14: Aproximación *Wavenet* de la Función $u(t) = -\frac{0.35}{100}t + cos(\frac{t}{17})e^{\frac{-t}{400}} + ruido$ con 20 Neuronas y *Wavelet* Morlet (a). La Actualización de los Parámetros Durante el proceso de Aprendizaje (b).

Tabla 3: Comportamiento de las *Wavenets* para diferentes valores de K.

	Iteraciones									
$K \backslash e$	0.500	0.350	0.250	0.200	0.150	0.100	0.030	0.020	0.010	0.001
5	0	4	6	8	10	13	30	66	> 300	> 300
10	0	3	4	5	6	7	13	17	27	148
15	0	2	3	4	5	6	9	12	19	100
20	0	2	3	4	5	6	7	9	15	77
30	0	2	3	4	5	6	7	8	11	56
45	0	2	3	4	5	6	7	8	9	40
60	0	2	3	4	5	6	7	8	9	32
80	0	2	3	4	5	6	7	8	9	25
100	0	2	3	4	5	6	10	11	14	29
120	0	40	41	42	43	45	52	54	59	76

Las Figuras 13 y 14 muestran los comportamientos de la aproximación *wavenet* implementando la *wavelet* madre Morlet con 20 neuronas hijas, con un umbral de error $\varepsilon = 0.001$ y un número máximo de iteraciones de 400 para las funciones $u(t) = e^{-t/50} sin(\frac{t}{5}) + ruido$ y $u(t) = -\frac{0.35}{100}t + cos(\frac{t}{17})e^{\frac{-t}{400}} + ruido$, respectivamente. Es importante comentar que el ruido que se añadió a las señales, es de tipo "*random*" acotado en el intervalo $[-\frac{1}{20}, \frac{1}{20}]$.

4. *Wavenets* con Estructura de Bloque IIR

Como se mencionó anteriormente, una *wavenet* es una red local en la que la función de salida está bien localizada en los dominios tanto del tiempo como de la frecuencia. Además, una red local doble puede ser conseguida combinando la arquitectura *wavenet* dispuesta en cascada con una red sináptica de respuesta al impulso infinita (IIR) local (Ye and Loh, 1993). El IIR de lazo recurrente crea una estructura local que provee un método computacionalmente eficiente de entrenamiento de la red, como resultado se obtiene un menor tiempo de convergencia en la aproximación de la señal. La Figura 15a muestra la estructura de la red para aproximar una señal $u(t)$, mediante la generalización de una combinación lineal de un conjunto de *wavelets* hijas, $h_{a,b}(t)$ dispuestas en cascada con el filtro IIR. La señal que es aproximada por la red, $\hat{y}(t)$ es modelada por:

$$\hat{y}(t) = \sum_{i=0}^{M} c_i z(t-i)u(t) + \sum_{j=1}^{N} d_j \hat{y}(t-j)v(t) \tag{12}$$

donde

$$z(t) = \sum_{k=1}^{K} w_k h_{a_k,b_k}(t) \tag{13}$$

K es el número de *wavelets*, w_k es el k-ésimo peso, M es el número de coeficientes de realimentación c_i del filtro IIR, mientras que N es el número de coeficientes de retroalimentación d_j de dicho filtro. La señal $u(t)$ es la entrada a ser aproximada y $v(t)$ es una señal persistente.

Los parámetros que se actualizan de la *wavenet*-IIR son: w_k, a_k, b_k, c_i y d_j, que son optimizados empleando el algoritmo del gradiente descendiente minimizando la función de energía del error, E definida como en (3), en los instantes del tiempo t.

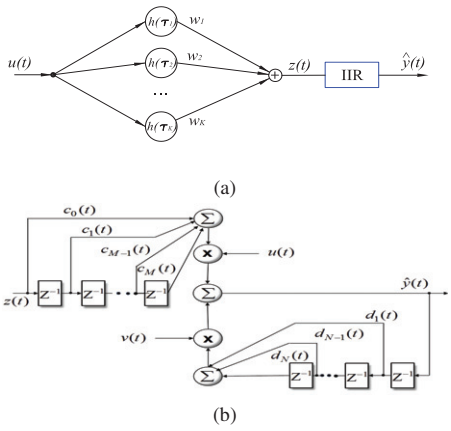


Figura 15: Estructura de Red *Wavenet*-IIR (a). Estrutura del filtro IIR (b).

Los gradientes $\frac{\partial E}{\partial w_k}$, $\frac{\partial E}{\partial a_k}$ y $\frac{\partial E}{\partial b_k}$ de los parámetros de la red *wavenet*, y $\frac{\partial E}{\partial c_i}$ y $\frac{\partial E}{\partial d_j}$ de los coeficientes de la estructura IIR requeridos para la minimización de E pueden ser expresados como:

$$\frac{\partial E}{\partial w_k} = -\sum_{t=1}^{T} e(t)u(t) \sum_{i=0}^{M} c_i h(\tau_k - i) \tag{14}$$

$$\frac{\partial E}{\partial b_k} = -\sum_{t=1}^{T} e(t)u(t) \sum_{i=0}^{M} c_i w_k \frac{\partial h(\tau_k - i)}{\partial b_k} \tag{15}$$

$$\frac{\partial E}{\partial a_k} = -\sum_{t=1}^{T} e(t)u(t) \sum_{i=0}^{M} c_i w_k \tau_k \frac{\partial h(\tau_k - i)}{\partial b_k} \tag{16}$$

$$= \tau_k \frac{\partial E}{\partial b_k}$$

$$\frac{\partial E}{\partial c_i} = -\sum_{t=1}^{T} e(t)u(t)z(t-i) \tag{17}$$

$$\frac{\partial E}{\partial d_j} = -\sum_{t=1}^{T} e(t)v(t)\hat{y}(t-j) \tag{18}$$

De manera similar, los cambios incrementales de cada parámetro son el negativo de sus gradientes:

$$\Delta \boldsymbol{w} = -\frac{\partial E}{\partial \boldsymbol{w}},\ \Delta \boldsymbol{a} = -\frac{\partial E}{\partial \boldsymbol{a}},\ \Delta \boldsymbol{b} = -\frac{\partial E}{\partial \boldsymbol{b}},\ \Delta \boldsymbol{c} = -\frac{\partial E}{\partial \boldsymbol{c}},\ \Delta \boldsymbol{d} = -\frac{\partial E}{\partial \boldsymbol{d}} \quad (19)$$

de este modo, el vector de cada coeficiente $\boldsymbol{w}$, $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$ y $\boldsymbol{d}$ de la red *wavenet* es actualizado por (9), (10) y (11), y para los parámetros del filtro IIR se emplean las reglas:

$$\boldsymbol{c}(t+1) = \boldsymbol{c}(t) + \mu_c \Delta \boldsymbol{c} \quad (20)$$

$$\boldsymbol{d}(t+1) = \boldsymbol{d}(t) + \mu_d \Delta \boldsymbol{d} \quad (21)$$

donde, μ son parámetros fijos que se determinan a prueba y error.

Algoritmo 2. El algoritmo *wavenet*-IIR que resulta es:

1. Calcular las salidas de la *wavenet*-IIR $\hat{y}(t)$ como en (12) para $t = 1, 2, ..., T$, es decir, una iteración (época).
2. Para cada uno de los valores de t calcular el error $e(t)$ con respecto a la entrada $u(t)$ definido en (4).
3. Obtener la función de energía del error E definida en (3) y calcular $\frac{\partial E}{\partial w_k}$, $\frac{\partial E}{\partial a_k}$, $\frac{\partial E}{\partial b_k}$, $\frac{\partial E}{\partial c_i}$ y $\frac{\partial E}{\partial d_j}$ dados por (14), (15), (16), (17) y (18), respectivamente.
4. Definir los incrementos $\Delta \boldsymbol{w}$, $\Delta \boldsymbol{a}$, $\Delta \boldsymbol{b}$, $\Delta \boldsymbol{c}$ y $\Delta \boldsymbol{d}$ para los parámetros $\boldsymbol{w}$, $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$ y $\boldsymbol{d}$ como en (19).
5. Se realizan las actualizaciones de los parámetros $\boldsymbol{w}$, $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$ y $\boldsymbol{d}$ de acuerdo con (9), (10), (11), (20) y (21).
6. Repetir el procedimiento el número de iteraciones (épocas) necesarias para que el error sea mínimo o alcance algún umbral dado $\varepsilon > 0$.

Tabla 4: Valores iniciales propuestos de los parámetros de la *wavenet*-IIR.

μ_w	μ_a	μ_b	w_k	a_k	b_k	μ_c	μ_d
0.01	0.001	0.0001	0	10	$\frac{T}{K}k$	0.02	0.02

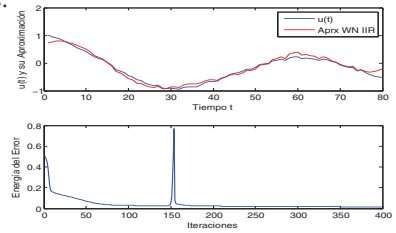
5. Aproximación de señales mediante una *Wavenet*-IIR

De la misma manera que en la Sección 3 se hace un análisis de la aproximación de algunas funciones mediante las *wavenets* pero ahora incluyendo la estructura del filtro IIR, mostrada en la Figura 15, donde se modifica el número de neuronas usadas y las *wavelets* implementadas en la red dadas en la Tabla 1, manteniendo en todos los casos un umbral mínimo del error de $\varepsilon = 0.001$ y modificando también el número de los coeficientes de la estructura IIR los realimentados c_i y los retroalimentados d_j.

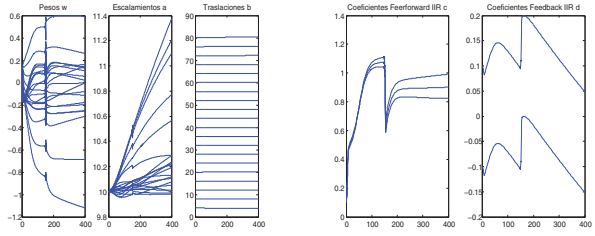
5.1. *Implementación de Diferentes Wavelets en la Aproximación de Señales*

A continuación se muestra el desempeño de la aproximación *wavenet*-IIR cuando se cambian las *wavelets* implementadas en la red mostradas en la Tabla 1, donde se mantiene fijo el número de neuronas de la red y los coeficientes del filtro IIR ($M = 2$ y $N = 2$), se fija un máximo de 400 iteraciones y el umbral del error de aproximación $\varepsilon = 0.001$, en todas las simulaciones los parámetros de la *wavenet*-IIR se inician con los mismos valores.

La Figura 16 muestra una *wavenet*-IIR con la *wavelet* SLOG1. Se observa que la función de energía del error sufre un cambio súbito alrededor de las 150 iteraciones y después disminuye, este cambio también se refleja en el comportamiento de los parámetros de la red, como en los del filtro IIR, esto se observa en las simulaciones mostradas en las Figuras 16b y 16c. Sin embargo, la señal aproximada no se altera. También se observa que el umbral de error no es alcanzado al cabo de las 400 iteraciones.



(a)



(b) (c)

Figura 16: Aproximación *Wavenet*-IIR con 20 Neuronas y *Wavelet* SLOG1 y la Energía del Error (a). La Actualización de los Parámetros de la Red Durante el Proceso de Aprendizaje (b) y del Filtro IIR (c).

La Figura 17 muestra la aproximación de la señal $u(t)$ con una *wavenet*-IIR con la wavelet RASP1. Se observa en la Figura 17a que el descenso de la función de energía del error es relativamente lento en comparación con los vistos en las aproximaciones usando la *wavelet* SLOG1, aunque éste sí alcanza niveles más bajos que los alcanzados en el caso de la *wavelet* SLOG1; pero no llega al umbral de error establecido.

La Figura 18 muestra una *wavenet*-IIR con la *wavelet* Morlet. El descenso de la función de energía del error, es más rápido que para los casos de las *wavelets* SLOG1 y RASP1 como se observa en la Figura 18a, alcanzando el umbral de error en 260 iteraciones.

Se observa que la *wavelet* que mejor desempeño proporciona es la Morlet, la cual es la única que alcanza el umbral mínimo fijado en menos de las 400 iteraciones, así como también es la que presenta un descenso mayor de la energía del error durante las primeras 50 iteraciones. Por esta razón en la siguiente subsección se presenta un estudio comparativo con esta *wavelet* haciendo una variación en el número de neuronas.

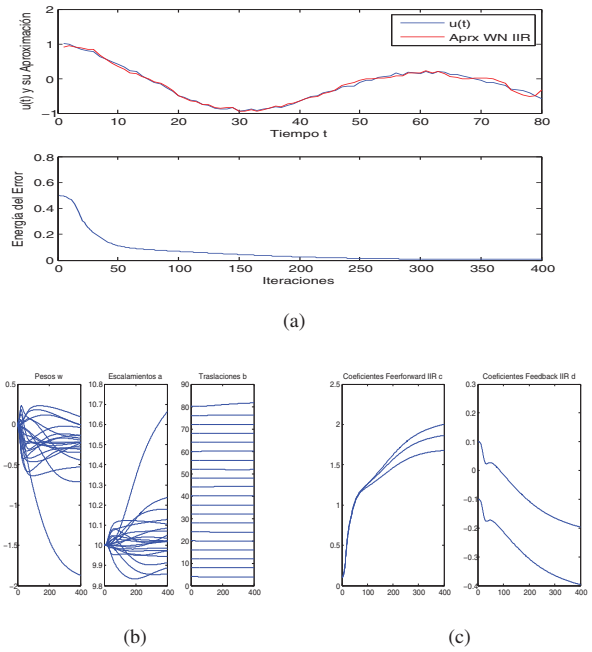


Figura 17: Aproximación *Wavenet* IIR con 20 Neuronas y *Wavelet* RASP1 y la Energía del Error (a). La Actualización de los Parámetros de la Red Durante el Proceso de Aprendizaje (b) y del Filtro IIR (c).

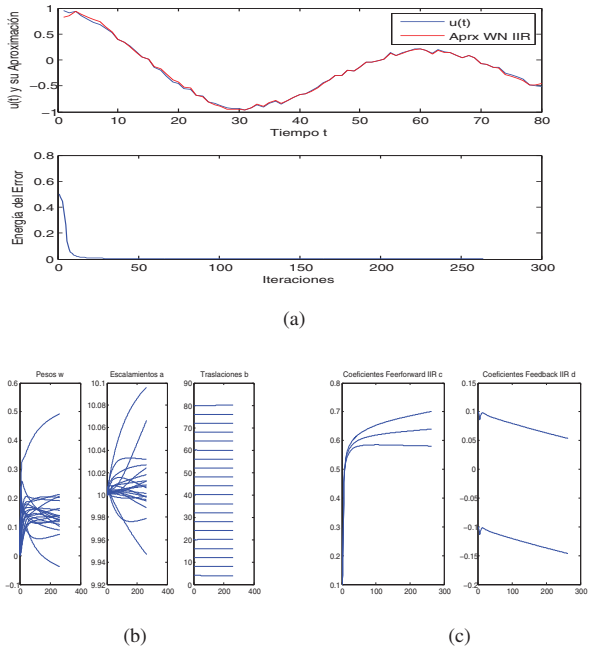


Figura 18: Aproximación *Wavenet*-IIR con 20 Neuronas y *Wavelet* Morlet y la Energía del Error (a). La Actualización de los Parámetros de la Red Durante el Proceso de Aprendizaje (b) y del Filtro IIR (c).

5.2. *Aproximación Wavenet-IIR con Wavelet Morlet y Diferentes Números de Neuronas*

En la Figura 19 se muestra el comportamiento de la aproximación de una señal con un número de 12 neuronas, con $M = 3$ y $N = 2$ para los coeficientes del filtro IIR.

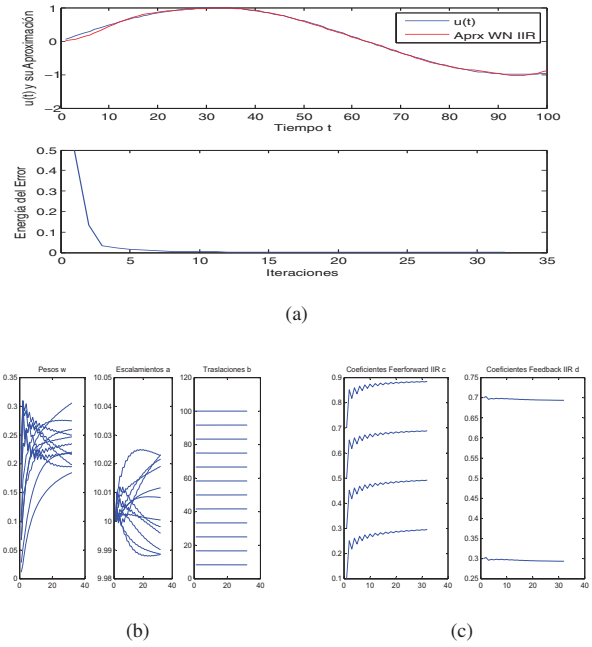


Figura 19: Aproximación *Wavenet*-IIR con 12 Neuronas y *Wavelet* Morlet y la Energía del Error (a). La Actualización de los Parámetros de la Red Durante el Proceso de Aprendizaje (b) y del Filtro IIR (c).

De la Figura 19a, se puede ver que el umbral de error es alcanzado en 32 iteraciones, que es menor comparado con el número de iteraciones que requirió la *wavenet* analizada en la Sección 3.2 cuyos resultados se mostraron en el Tabla 3, también se puede observar que la función de energía del error rápidamente alcanza valores bajos, los cuales también se pueden observar en la Tabla 5. En la Figura 19b se observa el comportamiento de los parámetros de la *wavenet*, mientras que en la Figura 19c se tiene el comportamiento de los coeficientes de realimentación y de retroalimentación del filtro IIR.

La Tabla 5 contiene algunos de los resultados de la aproximación de una función, implementando una *wavenet*-IIR en la que se modifican el número de neuronas y se obtienen los datos del número requerido de iteraciones para alcanzar algunos valores del error, incluyendo el umbral fijado. Esto es realizado sólo para valores de K que presentan un buen desempeño.

De esto se puede observar que la *wavenet*-IIR con mejor comportamiento en cuanto al número de iteraciones requeridas, es la que implementa 12 neuronas, alcanzando el umbral en 32 iteraciones. Estos resultados se obtuvieron para una *wavenet*-IIR cuyos parámetros iniciales se muestran en el Tabla 6. Se concluye que la *wavelet* que mejores resultados arroja es la

Tabla 5: Comportamiento de las *Wavenets* para diferentes valores de K.

	Iteraciones									
$K \backslash e$	0.500	0.350	0.250	0.200	0.150	0.100	0.030	0.020	0.010	0.001
10	0	2	3	4	5	6	7	8	9	63
11	0	2	3	4	5	6	7	8	9	36
12	0	2	3	4	5	6	7	8	9	32
13	0	2	3	4	5	6	7	8	9	57

Tabla 6: Valores iniciales y finales de los parámetros de la *Wavenets*-IIR mostrada en la Figura 19.

Wavenet-IIR									
Iniciales					Finales				
w	***a***	***b***	***c***	***d***	***w***	***a***	***b***	***c***	***d***
0	10	8.3333	0.1	0.3	0.3059	9.9902	8.3074	0.2961	0.2936
0	10	16.6666	0.3	0.7	0.1985	10.0232	16.6414	0.4925	0.6936
0	10	24.9999	0.5	–	0.2209	9.9959	25.0008	0.6887	–
0	10	33.3333	0.7	–	0.2353	9.9982	33.3353	0.8847	–
0	10	41.6666	–	–	0.2166	10.0217	41.6860	–	–
0	10	49.9999	–	–	0.2471	10.0005	50.0133	–	–
0	10	58.3333	–	–	0.1847	10.0083	58.3329	–	–
0	10	66.6666	–	–	0.2181	10.0117	66.6668	–	–
0	10	74.9999	–	–	0.2750	9.9885	74.9788	–	–
0	10	83.3333	–	–	0.1957	10.0229	83.3141	–	–
0	10	91.6666	–	–	0.2503	10.0191	91.6979	–	–
0	10	100	–	–	0.2602	9.9886	100.0031	–	–

Morlet, comparada con las otras *wavelets* analizadas en la Sección 3.

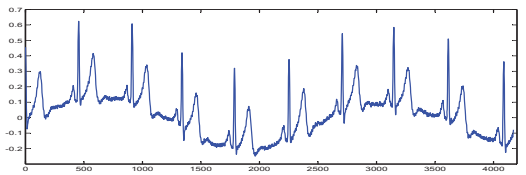
6. Comparación entre la Aproximación *Wavenet* y la *Wavenet*-IIR para una Señal ECG

En esta sección se compara el comportamiento de la aproximación cuando se implementa una *wavenet* contra una *wavenet*-IIR, para este estudio se trabaja con la red *wavelet* Morlet en ambos casos, esto se hace para una señal de un ECG la cual se muestra en la Figura 20 y fue obtenida de WEB (2009).

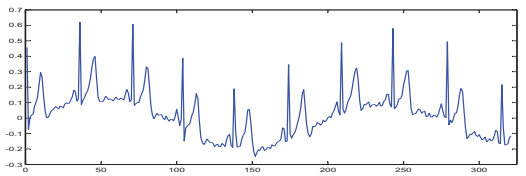
La señal original consta de un total de 4200 muestras la cual se observa en la Figura 20a, para efectos prácticos de prueba de convergencia de los algoritmos *wavenets* solo se utilizan 322 del total, dicha señal se observa en la Figura 20b. En cada caso, se realiza la aproximación para un máximo de 400 iteraciones o un umbral del error mínimo de $\varepsilon = 0.001$, en una red de 20 neuronas, los valores iniciales de los parámetros de la red son inicializados en los mismos valores en ambos casos y para la *wavenet*-IIR se usa un número de $M = 2$ y $N = 2$ para los coeficientes del filtro IIR.

En la Figura 21 se muestra el comportamiento de la aproximación para una *wavenet* del ECG de la Figura 20b. Se observa que la aproximación no alcanza el umbral del error antes de las 400 iteraciones, así como que la velocidad con la que desciende la energía del error es relativamente lenta.

En la Figura 22 se muestra el comportamiento de la aproximación para una *wavenet*-IIR del ECG de la Figura 20b. Se



(a)



(b)

Figura 20: Señal de un ECG (a). Muestras de la Señal del ECG (b).

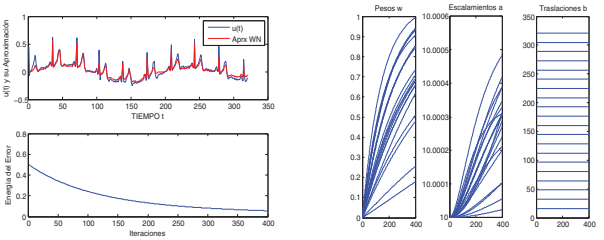
observa que la aproximación no alcanza el umbral del error antes de las 400 iteraciones, pero la energía del error desciende rápidamente con respecto de la que existe en la aproximación *wavenet*.

7. Comentarios

Las *wavenets* y *wavenets*-IIR son una buena herramienta en la aproximación de señales, mostrando un buen desempeño.

Tabla 7: Valores iniciales y finales de los parámetros de la *Wavenet* y de la *Wavenets*-IIR mostradas en la Figura 21 y 22, respectivamente.

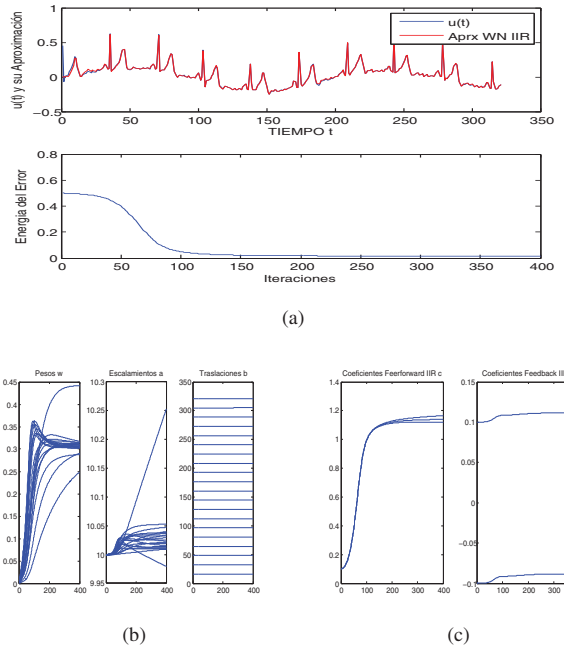
Iniciales					Finales							
Wavenet y Wavenet-IIR					*Wavenet*			*Wavenet-IIR*				
w	***a***	***b***	***c***	***d***	***w***	***a***	***b***	***w***	***a***	***b***	***c***	***d***
0	10	16.1	0.1	0.1	1.1095	10.0013	16.0380	0.4300	10.7267	15.3363	1.2167	0.1084
0	10	32.2	0.1	−0.1	1.0243	10.0002	32.1056	0.2984	9.9128	32.2358	1.1390	0.1084
0	10	48.3	0.1	–	1.0562	10.0006	48.1451	0.3119	10.0613	48.0889	1.0766	–
0	10	64.4	–	–	1.0298	10.0005	64.2045	0.3079	10.0231	64.2390	–	–
0	10	80.5	–	–	1.0323	10.0004	80.2438	0.3019	10.0202	80.2118	–	–
0	10	96.6	–	–	0.5132	10.0002	96.3023	0.3080	10.0180	96.3326	–	–
0	10	112.7	–	–	1.0175	10.0007	112.3469	0.3088	10.0548	112.3075	–	–
0	10	128.8	–	–	1.0376	10.0005	128.4008	0.3063	10.0151	128.4059	–	–
0	10	144.9	–	–	0.9892	10.0005	144.4499	0.3112	10.0332	144.4573	–	–
0	10	161.0	–	–	1.0580	10.0006	160.4989	0.3059	10.0228	160.4780	–	–
0	10	177.1	–	–	0.9916	10.0004	176.5512	0.3085	10.0495	176.5980	–	–
0	10	193.2	–	–	0.4104	10.0001	192.5990	0.3027	10.0111	192.5760	–	–
0	10	209.3	–	–	1.0322	10.0006	208.6561	0.3033	10.0397	208.6869	–	–
0	10	225.4	–	–	0.9717	10.0005	224.6950	0.3105	10.0162	224.6748	–	–
0	10	241.5	–	–	01.0595	10.0007	240.7563	0.3024	10.0545	240.7889	–	–
0	10	257.6	–	–	0.9417	10.0003	256.7945	0.3089	10.0104	256.7712	–	–
0	10	273.7	–	–	0.8169	10.0003	272.8547	0.3087	10.0157	272.8775	–	–
0	10	289.8	–	–	0.9868	10.0007	288.8961	0.3044	10.0358	288.8820	–	–
0	10	305.9	–	–	0.9777	10.0005	304.9527	0.3041	10.0420	304.9681	–	–
0	10	322.0	–	–	0.8164	10.0002	320.9965	0.2886	10.0094	320.9788	–	–



(a) (b)

Figura 21: Aproximación *Wavenet* de un ECG y la Energía del Error (a). La Actualización de los Parámetros de la Red Durante el Proceso de Aprendizaje (b).

Aunque es evidente que la estructura IIR dota a la *wavenet* de un mejor desempeño con respecto al número de iteraciones requeridas para alcanzar un umbral de error fijado o simplemente minimizar la función de energía en la aproximación de señales, ya sean señales aleatorias acotadas, señales algebraicas (señales que representan funciones algebraicas) o señales médicas (como el caso del ECG). Así mismo se concluye que para el caso de las señales aproximadas, la *wavelet* que mejor se comportó al ser implementada en los algoritmos *wavenet* y *wavenet*-IIR, es la *wavelet* Morlet. Una de las aplicaciones en la automática de los resultados presentados en este artículo se dan en la identificación y control de plantas estables como se muestran en (Cruz-Tolentino et al., 2010; Sedighizadeh and Rezazadeh,



(a)

(b) (c)

Figura 22: Aproximación *Wavenet*-IIR de la señal ECG y la Energía del Error (a). La Actualización de los Parámetros de la Red Durante el Proceso de Aprendizaje (b) y del Filtro IIR (c).

2008; Islas-Gómez et al., 2010, 2012).

8. Conclusiones

En este trabajo de investigación se aplican métodos adaptables en el diseño de algoritmos computacionales, dichos algoritmos emplean redes neuronales y series de *wavelets* para construir "neuroaproximadores" *wavenets*. Los algoritmos obtenidos se aplican en la aproximación de señales que representan funciones algebraicas y funciones aleatorias, así como en una señal médica de un electrocardiograma (ECG). De lo cual se puede concluir lo siguiente:

1. Que las *wavenets* presentan un buen desempeño en la aproximación de funciones, y que gracias a la multiresolución que tienen como característica nos permite ver los detalles finos de las señales involucradas.
2. La aplicación de las *wavenets* al campo de aproximación de señales promete una alternativa a los métodos clásicos sintonizables.
3. Los resultados arrojados de las simulaciones muestran el buen desempeño que tiene el aproximador *wavenet*-IIR.

El fenómeno del cambio súbito que se presenta en el esquema de aproximación de funciones propuesto, se observa que el error va disminuyendo y en una iteración determinada sube, eso indica que el método del gradiente descendiente no trabaja de forma eficiente, debido a que la función de error es muy compleja, esto se ve en la Figura 16a. Una alternativa es emplear otros tipos de entrenamiento como son: algoritmo LMS con error codificado, algoritmo mínimos cuadrados promediados de paso variable con error codificado, entre otros.

English Summary

Wavenet Algorithms with Applications in Approximation Signals: A Comparative Study.

Abstract

In this paper adaptable methods for computational algorithms are presented. These algorithms use neural networks and wavelet series to build neuro wavenets approximators. The algorithms obtained are applied to the approximation of signals that represent algebraic functions and random functions, as well as a medical EKG signal. It shows how wavenets can be combined with auto-tuning methods for tracking complex signals that are a function of time. Results are shown in numerical simulation of two architectures of neural approximators wavenets: the first is based on a wavenet with which they approach the signals under study where the parameters of the neural network are adjusted online, the other neuro approximator scheme uses an IIR filter to the output of wavenet, which serves to filter the output, in this way discriminate contributions of neurons that are less important in the approximation of the signal, which helps reduce the convergence time to a desired minimum error.

Keywords:
Signal processing, Self-adapting algorithms, Neural networks, Approximation algorithms, Gradient methods.

Referencias

Chen, D., Hui-Qiang, 2007. Approaches to realize high precision analog-to-dogital comverter based on wavelet neural network. In: International Conference on Wavelet Analysis and Pattern Recognition, Beijing, China.

Chui, C., 1992. An Introduction to Wavelets. Academic Press Inc, Boston, MA.

Cruz-Tolentino, J., Ramos-Velasco, L., Espejel-Rivera, M., 2010. A self-tuning of a wavelet PID controller. IEEE International Conference on Electronics Communications and Computers, 23–31.

Daubechies, I., 1992. Ten lectures on waveletes. CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM.

Gupta, M. M., Jin, L., Homma, N., 2003. Static and Dynamic Neural Networks: From Fundamentals to Advanced Theory. John Wiley and Sons.

Haykin, S., 1994. Neural Networks: A Comprehensive Foundation. Prentice Hall PTR, Upper Saddle River, NJ.

Haykin, S., 2001. Kalman Filtering and Neural Networks. John Wiley & Sons, New York.

Islas-Gómez, O., Ramos-Velasco, L., García-Lamont, J., 2010. Identificación y control wavenet de un motor de cd. Congreso Anual de la Asociación de México de Control Automático(AMCA), Puerto Vallarta, Jalisco, Mexico.

Islas-Gómez, O., Ramos-Velasco, L., Ramos-Fernández, J., García-Lamont, J., Espejel-Rivera, M., 2012. Identificación y control wavenet de un motor de ca. Revista Iberoamericana Automática e Informática (RIAI), en revisión.

Jun, W., Hong, P., 2005. Constructing fuzzy wavelet network modeling. International Journal of Information Technology 11, 68–74.

Kobayashi, K., Torioka, T., 1994. A wavelet neural network for function approximation and network optimization. In: Intelligent Engineering Systems Through Artificial Neural Networks, Volume 4, C. H. Dagli, B. R. Fernandez, J. Ghosh, and R. T. Soundar Kumara, Eds., Proceedings of the Artificial Neural Networks in Engineering (ANNIE ´94) Conference.

Li, S., Chen, S., 2002a. Function approximation using robust wavelet neural networks. In: 14th IEEE International Conference on Tools with Artificial Intelligence.

Li, S.-T., Chen, S.-C., Nov. 2002b. Function approximation using robust wavelet neural networks. 14th IEEE International Conference on Tools with Artificial Intelligence, 2002. Proceedings (ICTAI 2002)., 483–488.

Mallat, S., 1995. Wavelet Signal Processing. Academic Press.

Park, J., Sandberg, I. W., June 1991. Universal approximation using radial-basis-function networks. Neural omputation 3, 246–257.

Rousseeuw, P. J., Leroy, A. M., 1987. Robust Regression and Outlier Detection. Wiley.

S. Gopinath, I. K., Bhatt, R., 2004. Online system identification using wavelet neural networks. In: TENCON 2004. 2004 IEEE Region 10 Conference.

Sedighizadeh, M., Rezazadeh, A., 2008. Adaptive PID control of wind energy conversion systems using RASP1 mother wavelet basis function network. Proceeding of World Academy of Science, Engineering and Technology, 269–273.

Sitharama, S., Cho, E. C., Phoha, V. V., 2002. Fundations of Wavelet Networks and Applications. Chapman and Hall/CRC, USA.

Teolis, A., 1998. Computational Signal Processing with Wavelets. Birkhäuser, USA.

Ting, W., Sugai, Y., Oct. 1999. A wavelet neural network for the approximation of nonlinear multivariable function. IEEE International Conference on Systems, Man, and Cybernetics, IEEE SMC '99 Conference Proceedings. 3, 378–383.

Vetterli, M., Kovačević, J., 1995. Wavelets and Subband Coding. Prentice-Hall, USA.

W. Wang, T. Lee, C. L., Wang, C., 1997. Function approximation using fuzzy neural networks with robust learning algorithm. IEEE transactions on systems man and cybernetics Part B Cybernetics 27 (4), 740–747.

WEB, P., 2009. www.physionet.org/.

Widrow, B., Stearns, S. D., 1985. Adaptive Signal Processing. Prentice Hall, Inc., Englewood Cliffs, NJ.

Ye, X., Loh, N. K., 1993. Dynamic system identification using recurrent radial basis function network. In: Proceedings of American Control Conference.

Yu, W., Li, X., June 2003. Fuzzy neural modeling using stable learning algorithm. Proceedings of the American Control Conference Denver, Colorado, 4542–4548.

Zhang, Q., Beneveniste, A., 1992. Wavelet networks. IEEE Transactions on Neural Networks 3 (6), 889–898.